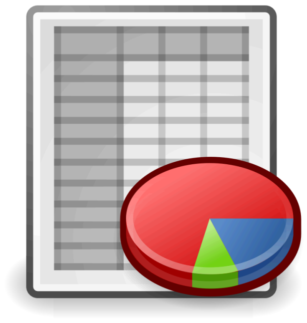
Office Automation 2.0

How can we make spreadsheets more robust as a business solution? We'll provide a case study with a working demo using the newest spreadsheet paradigm: Google Apps plus Google Apps Script.

Spreadsheets are not just a utility, they are a foundation of organization, calculation, work and analysis. But spreadsheets have limits. It takes technical expertise (not just a power-user) to ensure that the needs of the business are being solved by these fundamental tools. Traditionally, when technical expertise was available, programmers would use VBA (Visual Basic for Applications) to create desktop applications out of Office tools. These office applications were and are good intermediates between "simple" spreadsheets and full software programs. Then the Internet Age arrived and web solutions became more frequently prescribed over the desktop-constrained office application. Web solutions were great because the same set of technical skills and infrastructure could be used to solve a much greater range of problems and meet the needs of virtually every possible audience. You could definitely say that web solutions caused a near extinction of the VBA-powered office application. However, the ubiquity of web solutions did not cause spreadsheets to disappear. Today, spreadsheets are still the lingua franca of the office environment in companies big and small. Nor, did online shareable spreadsheets magically solve traditional limitations of spreadsheets. We'll see in a moment what is now available as a very powerful adjunct to the "simple" spreadsheet.

Today, spreadsheets have moved off the desktop into the cloud. Offerings such as Google Apps provide spreadsheets over the Internet fulfilling the promise of applications that are available everywhere to everyone to every challenge. But spreadsheets online, in and of themselves, do not eliminate the limitations of spreadsheets as full-fledged business solutions. The answer? Google Apps Script (GS). GS is the VBA equivalent for the Internet Age. It marries JavaScript, already established as a core component of the web, with the full product portfolio of Google Apps plus external libraries, APIs and data sources. This means that companies can now use their web expertise to add business rules, logic, automation, User Interfaces and other application features to their spreadsheets when their needs have outgrown the seductive simplicity of the ubiquitous spreadsheet.



It should be obvious I'm talking about doing what's possible, given the technical knowledge of what's possible. I'm not talking about taking off the shelf products, plugging them in, turning them on and forgetting about them. But then again, the ambitious user with some level of technical competence can achieve great results with speed and a level of responsiveness and customization that just wasn't possible earlier. Let's take a look at an example case study.

Case Study

Let's imagine a company with many office locations and nearly 1000 service agents in the field, the agent phone list becomes a pretty big spreadsheet. And people use the spreadsheet for lots of different purposes[1]. You want the data all centralized, but sometimes you want just the list of agents affiliated with a particular office. Our simple case study answers the question: How do you let multiple people share the same spreadsheet data[2] while also protecting the data from accidental loss due to an inadvertent keystroke just because the spreadsheet is open for viewing? You could create a copy of the master data, and "protect" the copy by making it read-only, but then how do you synchronize the data, and ensure that other people viewing the live data are actually seeing the most updated information in the "read-only" copy? You can do all of this in Google Apps.

Demo

The sample phone list spreadsheet has over 900 example agents plus their contact info. Additions to the phone list, record deletions, or phone number changes are the most common events in our case study. Accordingly, the sample phone list spreadsheet uses GS code[3] to automatically and instantly update the protected “Read Only” sheet whenever changes are made to the “**Master**”.

The Agents by Office sheets are created programmatically; eliminating manual data duplication, errors and omissions. What’s more, this automation is attached to an “on open” event which re-creates all the “Agents by Office” sheets every time the spreadsheet is opened to ensure that the data is fresh. In practice, these reference lists do not need to be updated instantaneously for every minute change since a change in office affiliation is a rare event compared with add/delete/edit of a phone record. As a further example of the strength of the GS platform, a custom menu option has been programmed providing the power to generate the “Agents by Office” sheets on demand.

You can learn more about GS at script.google.com

About the Author

Gregory Scott Rundlett is a longtime Free Software advocate participating in the Boston Linux User Group and Greater New Hampshire Linux User Group. He lives with his two sons in Salisbury, MA. He is also founder of Equality Tech (http://equality-tech.com/), a solutions company. When you talk with us, we involve leadership and all constituents to discover goals, current state of practice and performance, workflow, and the infrastructure underlying these initiatives to produce technology solutions that improve your results.

skype/aim/irc/twitter freephile | voice 978-518-7601 | greg@rundlett.com

[1] You could of course use a Directory Service such as OpenLDAP to organize people, but the example company has not yet instituted any formal application for user management.

[2] Document permissions and sharing are built-in features of Google Drive, so that requirement is already taken care of.

[3] See the code at http://pastebin.com/0qVySpZr